AMENDMENTS TO THE SPECIFICATION

Please delete the previous version of the abstract of the disclosure. A substitute abstract is appended hereto as a separate page.

Amend the paragraph beginning on page 1, line 7, to read as follows:

a1

Communication networks are well known in the computer communications field. By definition, a network is a group of computers and associated devices that are connected by communications facilities or links. Network communications can be of a permanent nature, such as via cables, or can be of a temporary nature, such as connections made through telephone or wireless links. Networks may vary in size, from a local area network ("LAN") consisting of a few computers or workstations and related devices; to a wide area network ("WAN") which interconnects computers and LANs that are geographically dispersed; or to a remote access service ("RAS") which interconnects remote computers via temporary communication links. An internetwork, in turn, is the joining of multiple computer networks, both similar and dissimilar, by means of gateways or routers that facilitate data transfer and conversion from various networks. A well-known abbreviation for the term internetwork is "internet." As currently understood, the capitalized term "Internet" refers to the collection of networks and routers that use the Internet Protocol ("IP")U, along with higher level protocols such as the Transmission Control Protocol/Internet Protocol ("TCP/IP") or the Uniform Datagram Packet/Internet Protocol ("UDP/IP"), to communicate with one another.

Amend the paragraph beginning on page 2, line 20, to read as follows:

a2

A user is allowed to retrieve hypertext documents from the Web, i.e., a user is allowed to "surf the Web," via a Web browser. A Web browser, such as NETSCAPE NAVIGATOR®, MICROSOFT® ~~Internet Explorer~~ INTERNET EXPLORER or phone.com's UP.link

a2 cont

microbrowser, is a software program implemented by a Web client, i.e., the user's computer, cell phone or other client device, to provide a graphical user interface ("GUI") to the Web. Upon request from the user via the Web browser, the Web client accesses and retrieves the desired hypertext document from the appropriate Web server using the URL for the document and a protocol known as HyperText Transfer Protocol ("HTTP"). HTTP is a higher-level protocol than TCP/IP and is designed specifically for the requirements of the Web. It is used on top of TCP/IP to transfer hypertext documents between servers and clients.

Amend the paragraph beginning on page 2, line 31, to read as follows:

a3

At the advent of the Web, the information stored on the Internet was generally static in nature and if one wanted to change the information provided on a Web site it was necessary to manually configure the Web site by rewriting its HTML code. However, at the present stage of development on the Web, many Web sites provide dynamic content that changes depending on a user's interaction between the Web browser on the user's client device and the Web site. These dynamic hypertext documents are well known in the art and may be produced in a myriad of different manners, such as by using Common Gateway Interface ("CGI") scripts processed by a Web server or local scripts just as JAVAScript is processed by a Web browser.

Amend the paragraph beginning on page 3, line 5, to read as follows:

a4

The present invention relates to network-based, and Internet-based travel services, such as a travel service offering tickets for transportation, including airline tickets, train tickets, bus tickets, ferry tickets, etc., to customers over the Internet. With such a service, a customer, using a computer connected to the travel service via the Internet, can purchase items from a dynamically changing inventory including airline tickets, train tickets, bus tickets, ferry tickets, etc., or combinations thereof. Typically, a travel service cooperates with a centralized computer

A4 Cont

reservation system ("CRS"). A CRS is a system/service that communicates with travel agents, transportation providers or services for the purpose of providing up-to-date fare (price for the trip or combination of ~~routs~~ routes comprising a trip), schedule (date and time of arrival and departure of a trip or flight), rules (which fares are valid under which circumstances) and availability (capacity for a particular trip or flight to provide accommodation at a particular fare) in response to a query. This information is provided to the CRS by the transportation carriers, typically through third parties; however, a travel service can also cooperate with other databases, such as a local database reflecting specific relationships between carriers and the travel service, such as discount contracts or incentive programs. Accordingly, an Internet-based service can have access to many sources of carrier inventory and prices.

Amend the paragraph beginning on page 4, line 9, to read as follows:

A5

Furthermore, in a conventional Internet-based travel service, a consumer enters very specific information concerning desired supplier inventory, and the ~~internet-based~~ Internet-based travel service queries a CRS for inventory that matches that specific query. [[the]] The CRS performs a search of its database to find matches for the query and returns a result set to the Internet-based travel service for viewing by the consumer. However, a traditional CRS search in response to a query is limited.

Amend the paragraph beginning on page 5, line 15, to read as follows:

A6

The present invention solves the problem of finding best fares in a non-prohibitive manner by utilizing a query server with a branching and bounding based technique to implicitly enumerate possible travel solutions to arrive at best fare solutions. ~~By using an implicit enumeration, the~~

Amend the paragraph beginning on page 5, line 32, to read as follows:

a7

Adding trip information and eliminating partial fare solutions may be performed in a recursive or an iterative manner, so long as the possible solutions are at least implicitly examined. In fact, adding trip information and eliminating partial fare solutions may be [[the]] performed both backward and forward from a destination and origin to further decompose the search for best fares.

Amend the paragraph beginning on page 9, line 31, to read as follows:

a8

FIGURE 4 depicts several of the key components of the query server 400. Those of ordinary skill in the art will appreciate that the query server 400 includes many more components [[then]] than those shown in FIGURE 4. However, it is not necessary that all of these generally conventional components be shown in order to disclose an illustrative embodiment for practicing the present invention. As shown in FIGURE 4, the query server 400 is connected to the Internet 100 via a network interface 430. Those of ordinary skill in the art will appreciate that the network interface 430 includes the necessary circuitry for connecting the query server 400 to the Internet 100, and is also constructed for use with the TCP/IP protocol or the next generation protocols, such as the IIOP, the particular network configuration of the operating environment in which it is contained and a particular type of coupling medium.

Amend the paragraph beginning on page 10, line 8, to read as follows:

a9

The query server 400 also includes a processing unit 410, a display 440, and a mass memory 450 all interconnected along with the network interface 430 via a bus 420. The memory 450 generally comprises RAM, ROM, and one or more permanent mass storage devices, such as a hard disk drive, tape drive, optical drive, floppy disk drive, or combination

A9 cont

thereof. The mass memory 450 stores the program code and data necessary for receiving, processing, formatting and sending messages, as well as[[,]] supplying the results of that processing to senders in accordance with the present invention. More specifically, the memory 450 stores minimum cost matrices ("MCM") 465 which are used to store precalculated minimum cost fares between travel points (origins and destinations) in an easily accessible matrix format such that given an origin and a destination, it is possible to find the minimum cost of traveling between them. Additionally, the memory 450 stores a fare, schedule and rule database ("FSR") 470, a priority queue 475 and an operating system 455. The FSR is a collection of optimized fare, schedule and rule records that have been preprocessed as described below with regard to FIGURE 5. The fare records provide detailed trip information such as, but not limited to, the origin, destination, date, time and price of various trips. The ~~rules~~ rule records provide different rules (e.g., fare must be purchased 14 days in advance, fare not available if traveling via Denver, etc....) which may be associated with particular fares to more particularly specify when the fare is valid and how it may be used. The schedule records are more general records which specify the regularly scheduled trips provided by a carrier.

Amend the paragraph beginning on page 12, line 4, to read as follows:

A10

After all of the raw fare records received from the CRS 250 have been processed as described above, optimization continues in block 530 where the intermediate records are retrieved from the fare index 360. Then, each of the intermediate records are individually processed by first linking any records which have been identified as having links to at least one other record in block 535. These newly linked records are then stored as intermediate records, [[form]] forming further optimized fare records with explicit links to associated records. Next in block 540 a number of values are precalculated for the current intermediate record. For example, fare records add a list of all rule records to which they could link. This eliminates some runtime

a10 Cont

link processing associated with records to which the fare could never be linked, also, dollar cost per direction of travel is precalculated (i.e., half the fare level in the case of round trip fares) in order to make one way and round trip fares more directly comparable. It may also be useful to precalculate city codes that correspond to the airport codes (Three Letter Acronyms or "TLAs") in Block 540. Then block 545 recodes any values in the records to more easily process the record in the future. For example, dates are recoded to be represented as a number of days from a fixed base rather than as year, month and day format (YYDDMM). This allows for easier comparisons of dates as integers rather than strings which must first be converted before calculations may be performed on them. The record is then stored, in block 550 as an optimized fare record in the FSR 470 for later retrieval as either a fare record, a rule record or a schedule record. This process repeats with blocks 535, 540, 545 and 550 until decision block 555 determines that the last intermediate record from the fare index 360 has been processed. Then in block 560 the optimized fare records stored in the FSR 470 are examined for the lowest cost fares between any origin and destination points. These lowest cost fares are then stored in the MCM 465 for use by the query server 400 later in determining best fares. Processing then ends at block 599.

Amend the paragraph beginning on page 14, line 11, to read as follows:

a11

In understanding how the present invention is able to find the best fare(s) using only an implicit enumeration of possible solutions in an exemplary context, such as the air transportation context, rather than an explicit enumeration, it is helpful to understand the trip information used to describe and partition the possible solutions. FIGURE 13A represents a single trip from an origin at point A 1310 to a destination at point B 1330. The breakpoints (e.g., any point along the route, including the origin and destination) for this flight are at points A 1310, C 1320 and

B 1330. Point D 1315 is of some interest as it may also be an airport, but is not considered a breakpoint for purposes of determining carriers, flights or fares. It is merely a stopover point. The trip in FIGURE 13A is composed of two fares, a first flight between origin point A and breakpoint C with flight number "MA #100" 1335 that has a stopover at point D 1315; and a second nonstop flight between breakpoint C and destination point B with a flight number "HA #200" 1340. For purposes of the present invention, the "cost" of this [[a]] trip is composed of the actual time to get from point A 1310 to point B 1330 combined with the sum of the fares for the flights in between. The combination may be weighted to provide price or total travel time as more important. It will be appreciated by those of ordinary skill in the art that any number of weighting or ranking schemes may be used to determine a cost using price and travel time (possibly including or excluding [[lay-over]] layover time). It is possible that other information may be used in addition or in place of price and travel time. Carrier reliability and/or reputation may also be considered, for example.

a11 Cont

Amend the paragraph beginning on page 14, line 32, to read as follows:

a12

When determining fares for a trip, it is customary to determine to what type of priceable unit the fares will be applied. Most priceable units fall into one of three types of categories: one-way, round trip or open jaw. FIGURE 13C provides a simple example for illustrating the distinctions. A trip from 13J through 13K to 13L and back to ~~back to~~ 13J could be grouped into many possible priceable unit combinations. For example, in its simplest form, it is a round trip from 13J to 13L with a stopover at 13K on the outbound leg of the round trip. It could also be a group of three one-way flights, where each fare component 1380, 1385 and 1390 is a one-way priceable unit. Another grouping is the so called "open jaw" flight where the origin of the outbound leg and any subsequent leg(s) of a trip are not necessarily the same (but could be). In

A12 cont

one such example, fare components 1380 and 1385 could be part of an open jaw priceable unit, and fare component 1390 would be a one-way priceable unit. One way flights can comprise one or more fare components. Generally, round trip flights and open jaw ~~flight~~ flights are less expensive than one-way flights, so it is very beneficial to combine flights into these types of priceable units when trying to minimize the cost of a trip. Even from the simple example illustrated in FIGURE 13C, it is possible to see that with many breakpoints and only three types of priceable units, the possible permutations that would be added to a solution tree 1400 could quickly make an explicit enumeration of solutions impracticable.

Amend the paragraph beginning on page 15, line 25, to read as follows:

A13

In accordance with the present invention, a branch-and-bound technique is used to search for best fares by solving for a sequence of partial solutions which are derived from a query. For example, a first level of new partial solutions is created as nodes of a solution tree 1500 such that the partial solutions are only restrained by which breakpoints a trip must pass through~~, this~~. This results in a simpler problem which can be solved more easily with less computation. Branching splits a program into two sub-problems and bounding computes the lower bound (current best case) for each sub-problem. If the lower bound for the sub-problem is no better than the threshold cost, then the entire sub-problem (possible solution) is deferred until the threshold cost increases (if ever). A branch-and-bound process is discussed here as a tree, where every node in the tree represents a sub-problem (partial solution). Initially, the tree is initiated with a single root node which represents a relaxed problem where all variables in the non-root nodes of the tree are relaxed and replaced by the relaxed variable(s) in the root node. At each stage of the branch-and-bound search, one active node is selected and the associated relaxed problem is solved. Depending on the solution, one of the following three actions is taken:

~~Defering~~ Deferring: If the relaxed problem has a solution that is worse than the current best feasible objective value, then defer processing the node.

a13 Cont

Refining: Update the lower bound if the solution is such that for each node the cost would be below the previous lower bound.

Branching: Branch on some node if the relaxed solution is not complete and if its estimated value is better than the current best feasible solution.

Amend the paragraph beginning on page 17, line 9, to read as follows:

a14

During one exemplary operation of best fare processing (described in much greater detail below with reference to FIGURES 6-12) partial solutions (solution tree nodes) are partitioned (as can be seen in the branched nodes in FIGURE 15) according to an arbitrary threshold cost. In one exemplary embodiment the threshold cost starts at $100 and one hour, but any other amount could be used so long as it ~~bares~~ bears some relation to expected travel costs, possibly as determined by the MCM 465. Each partial solution is evaluated using the MCM 465 to determine if it could feasibly provide a complete solution below the threshold cost. If it could, then further information is added (e.g., the current node is partitioned and nodes at a new level are added) and those new partial solutions are evaluated in turn. If not, then the partial solution is not processed (e.g., it is deferred such as in node 1590), and processing continues until enough complete solutions are found or the processing times out. If enough complete solutions have not been found, time is left, and some partial solutions have not been evaluated, the threshold cost is increased and the unevaluated nodes are revisited with the higher threshold. In one exemplary embodiment the threshold cost increases by $50 and half an hour, but any other amount could be used so long as it ~~bares~~ bears some relation to expected travel costs, possible as determined by the MCM 465. This continues until enough complete solutions (e.g., enough leaf nodes, which could be any predesignated number, the exemplary Web pages of FIGURES 17A-C provide two

A14 Cont

sets of three flights, giving a total of nine possible complete solutions) have been found, the solutions are exhausted, or the process times out. At which point, any complete solutions (up to an arbitrary number) are presented to the user (usually with the most optimal solutions presented first).

Amend the paragraph beginning on page 18, line 33, to read as follows:

A15

As illustrated in FIGURES 2, 4 and 6, the present invention comprises a query server 400 that is used to determine the best fares requested by a client device 210. A flow chart illustrating the best fare routine 700 implemented by the query server 400 to determine the best available fare(s) in accordance with one actual embodiment of the present invention is shown in FIGURE 7. The best fare routine 700 begins in block 701 and proceeds to block 705, where a fare query is received. The query could come from a myriad of sources, such as third party travel servers or carriers, but for exemplary purposes, one embodiment of the present invention has the query forwarded from a customer using a client device 210. The information in the query is used next in subroutine block 800 (described in more detail below with regard to FIGURE 8) to add fare breakpoints to partial solutions and add those partial solutions below a threshold cost to a solution tree 1500 stored in the priority queue 475. The logic continues to subroutine block 900 (described in more detail below in reference to FIGURE 9) where the fare components defined by the breakpoints found in block 800 are assigned carriers, if they form partial solutions below a threshold cost. These carrier assignments are added to the partial solutions in the priority queue 475. Once the carriers have been assigned, processing continues to subroutine block 1000 (described below in detail for FIGURE 10) where flights are assigned to fare components having carriers and that would form partial solutions below a threshold cost. These flight assignments update the partial solutions of the solution tree 1500 stored in the priority queue 475. Next, in subroutine block 1100 (described in more detail with regard to FIGURE 11) the fare components with flight assignments are grouped to form priceable units if the resulting partial solution is below a threshold cost. These priceable units are then used to update the partial solutions in the priority queue 475. Accordingly in subroutine block 1200 (described below with regard to FIGURE 12) it is then possible to assign fares to the priceable units where the resulting partial solution would be below the threshold value. The partial solutions in the priority queue 475 would also be updated with these assignments. The processing would continue to block 710 where any partial solutions with the same trip information (except for fare information) would be eliminated. Providing fare data on the same flights, but with the same or higher fare is not necessary. Then, in decision block 715 a determination is made whether an ending scenario has been reached (e.g., a predetermined set of complete solutions, all possibilities exhausted or the process has timed out beyond some predetermined time limit). If no ending scenario was determined to have been reached in decision block 715, then processing continues to block 725 where the threshold cost is increased by an arbitrarily selected predetermined amount and processing returns to subroutine block 800. If an ending scenario was reached in decision block 715, then processing continues to block 720 where the lowest cost solutions (each containing a complete trip description with one or more priceable units and associated fares) are dequeued from the priority queue and a response to the fare query is generated by retrieving (and totaling) the total price of fares from each lowest cost solution, along with the trip description for the lowest cost solutions. The final price and trip description are then formatted (such as in Web pages 1710, 1720 and 1730 of ~~FIGUR~~ FIGURES 17A-C) and sent to the requesting device. Routine 700 ends at block 799.

A15

Amend the paragraph beginning on page 20, line 9, to read as follows:

A16

As mentioned above, a number of subroutines are described in FIGURES 8-12. Each subroutine partitions and adds further information to the solution tree 1500 of possible solutions.

a16 cont

Before examining the subroutines in detail, it is helpful to consider the two solution trees 1400 and 1500 illustrated in FIGURES 14 and 15, respectively.

Amend the paragraph beginning on page 20, line 14, to read as follows:

a17

In FIGURE 14, as each additional piece of trip information is added to the solution tree 1400, at each level, the number of partial solutions exponentially increases. Each additional layer in this prior art example partitions the previous partial solutions into two more partial solutions. However, as will be shown in the following subroutines, a process of "~~defering~~ deferring" nodes from consideration when building the tree creates a much more manageable solution tree 1500. Only the solutions that are below a certain cost are considered and those with a cost that is too high are not considered (i.e., "deferred") for further processing until the threshold cost increases.

Amend the paragraph beginning on page 20, line 23, to read as follows:

a18

As shown in FIGURE 15, processing begins with a root node 1505 representing the fare query. No details have been added to any other nodes in the solution tree 1500 at this point. At level 1, partial solutions with fare breakpoints 1510 are added (see FIGURE 8) to the solution tree 1500, in example solution tree 1500 both breakpoints allow for partial solutions below the threshold cost. At level 2, carriers 1515 are assigned (see FIGURE 9), but only three of the four partial solutions meet the threshold cost. Next, at level 3 flights are assigned 1520 to fare components with assigned carriers (see FIGURE 10), and only three of the six partial solutions meet the threshold cost. In turn, level 4 fare components are assigned to priceable units 1525 (see FIGURE 11), and only five of the six partial solutions meet the threshold cost. Finally at level 5, the fares (including taxes and some changes) are assigned to the priceable units (see FIGURE 12), and only seven of the ten ~~partials~~ partial solutions meet the threshold cost. Only a18 cont'd

the level 5 nodes[[,]] represent the set of complete solutions 1530, so instead of thirty-two possible solutions as in FIGURE 14, there are only ten. It is a much simpler task then to select the lowest cost complete solutions (leaf nodes) from the solution tree 1500. In one embodiment of the present invention the selection is aided by storing the complete solutions in a priority queue 475 which allows the removal of the lowest cost solutions first.

Amend the paragraph beginning on page 21, line 10, to read as follows:

a19

Starting with FIGURE 8, subroutine 800 begins at block 801 and proceeds to block 805, where a new breakpoint (any airport present in the data may be added as a breakpoint) is retrieved from the MCM 465 and is sought to be added to a new unique partial solution in level 1 of a solution tree 1500, such that the new partial solution will not exceed a threshold cost (in one exemplary embodiment, the same threshold is used in the calling routine), the breakpoint does not duplicate one already in another partial solution and some carrier publishes fares and provides transportation from the previous breakpoint. For purposes of discussion of the present invention, exemplary solution tree 1500 will be used, however, one of ordinary skill in the art will appreciate that a solution tree used by an actual embodiment of the present invention may be of considerable more complexity and may possibly include more levels and/or nodes than the solution tree 1500 illustrated in FIGURE 15. Any previous partial solutions to which a breakpoint is to be added are retrieved from the priority queue 475. The cost of a potential new unique partial solution is determined using the costs in the preprocessed MCM 465 to determine what a minimum feasible cost will be. More specifically, the cost of a partial solution is calculated using all known trip information and then by using the MCM 465 to complete any unknown trip information with minimum cost feasible trip information. In block 805 this means that only the user query and any previously determined breakpoints are known trip information, so the MCM 465 would supply minimum cost feasible carrier, flight, priceable unit and fare information to a new unique possible solution. As each new piece of trip information is added to the partial solutions, the cost of the partial solution will be refined, and will possibly increase, thereby removing it from consideration if the increased cost exceeds the threshold. 919

Amend the paragraph beginning on page 22, line 9, to read as follows:

920 Subroutine 800 is used to populate level 1 of the solution tree 1500. Repeated calls to subroutine 800 will (if available) provide the possible solutions (i.e., level 1 nodes) that will then be used to build the next levels (i.e., level 2 nodes and so on) of the solution tree 1500. Level 1 is not built all at once by subroutine 800, rather, by iterations or recursions. ~~subroutine~~ Subroutine 800 is repeatedly called and incrementally adds possible solutions that are below the current threshold cost. However, once subroutine 800 has finished, it is then possible that a partial solution has been added that will provide a best fare, so routine 700 would then call the next subroutine ~~900 for building level 2 of the solution tree 1500.~~

Amend the paragraph beginning on page 22, line 18, to read as follows:

921 In FIGURE 9, the building of the solution tree 1500 continues at the next level by adding carrier information retrieved from the MCM 465. Subroutine 900 begins at block 901 and proceeds to block 905, which tries to find a carrier which publishes fares and provides transportation over the fare component to be added to a fare component to form a partial solution that will not exceed a threshold cost (in one exemplary embodiment, the same threshold is used in the calling routine) also using the MCM 465. The partial solutions with breakpoint assignments are retrieved from the priority queue 475. The cost of a potential new unique partial solution is determined using the costs in the preprocessed MCM 465 to determine what a minimum feasible cost will be. More specifically, the cost of a new partial solution is calculated using all known trip information and then by using the MCM 465 to complete any unknown trip

information with minimum cost feasible trip information. In block 905 this means that only the user query, breakpoint(s) and previously assigned carrier(s) are known trip information, so the MCM 465 would supply minimum cost feasible flight, priceable unit and fare information to a new unique possible solution. As each new piece of trip information is added to the partial solutions, the cost of the partial solution will be refined, and will possibly increase, thereby removing it from consideration if the increased cost exceeds the threshold. If such a carrier was not found in block 905, processing continues to block 999, where any partial solutions to which a carrier has been added are returned to the calling routine (in this case, the best fare routine 700). Otherwise, if a carrier was found, then a new unique partial solution is created by adding the carrier to a fare component retrieved ~~form~~ from a previous solution in block 915 and adding the partial solution to the solution tree 1500 stored in the priority queue 475. Processing then returns to block 905 to search for any more partial solutions in which a carrier may be added.

A21 Cont'

A22

Amend the paragraph beginning on page 23, line 9, to read as follows:

Similar to subroutine 800, subroutine 900 is used to populate level 2 of the solution tree 1500. Repeated calls to subroutine 900 will (if available) provide the carriers to possible solutions provide (i.e., level 2 nodes) that will then be used to build the next levels (i.e., level 3 nodes and so on) of the solution tree 1500. Subroutine 900 is ~~repeated~~ repeatedly called and incrementally adds solutions that are below the current threshold cost. However, once subroutine 900 has finished, it is then possible that a partial solution has been added that will provide a best fare, so routine 700 would then call the next subroutine 1000 for building level 3 of the solution tree 1500.

Amend the paragraph beginning on page 24, line 8, to read as follows:

A23

Similar to subroutines 800 and 900, subroutine 1000 is used to populate level 3 of the solution tree 1500. Repeated calls to subroutine 1000 will (if available) provide the flights the possible solutions (i.e., level 3 nodes) that will then be used to build the next levels (i.e., level 4 nodes and so on) of the solution tree 1500. Subroutine 1000 is ~~repeated~~ repeatedly called and incrementally adds solutions that are below the current threshold cost. However, once subroutine 1000 has finished, it is then possible that a partial solution has been added that will provide a best fare, so routine 700 would then call the next subroutine 1100 for building level 4 of the solution tree 1500.

Amend the paragraph beginning on page 24, line 17, to read as follows:

A24

FIGURE 11 continues to build the solution tree 1500 by grouping fare components into priceable units. Subroutine 1100 begins at block 1101 and proceeds to block 1105, which tries to find a partial solution that will not exceed a threshold cost (in one exemplary embodiment, the same threshold is used in the calling routine) where flight(s) have been assigned, but priceable units (retrieved from the FSR 470) have yet to be determined. Block 1105 uses the FSR 470 to determine which priceable units are available for determination. The partial solutions with flight assignments are retrieved from the priority queue 475. The cost of a potential new unique partial solution is determined using the costs in the preprocessed MCM 465 to determine what a minimum feasible cost will be. More specifically, the cost of a new partial solution is calculated using all known trip information and then by using the MCM 465 to complete any unknown trip information with minimum cost feasible trip information. In block 1105 this means that the user query, breakpoint(s), carrier(s) and flight(s) are known trip information, so the MCM 465 would supply minimum cost feasible fare information, while the FSR 470 would supply the actual priceable unit information to a new unique possible solution. As each new piece of trip information is added to the partial solutions, the cost of the partial solution will be refined, and will possibly increase, thereby removing it from consideration if the increased cost exceeds the threshold. If a partial solution below the threshold cost where priceable units have yet to be defined was not found in block 1105, processing continues to block 1199, where any partial solutions, in which a priceable units(s) has been determined, are returned to the calling routine. Otherwise, if a partial solution meeting the criteria was found, then new unique partial solution(s) are created by grouping (as described above with reference to FIGURE 13C) fare component(s) into priceable unit(s) in all possible combinations where the partial solutions are still below the threshold cost. These new partial solutions are then added to ~~and adding it to~~ the priority queue 475. Processing then returns to block 1105 to search for any more partial solutions in which priceable units have yet to be determined.

A24 cnt

Amend the paragraph beginning on page 25, line 10, to read as follows:

A25

Similar to subroutines 800, 900 and 1000, subroutine 1100 is used to populate level 4 of the solution tree 1500. Repeated calls to subroutine 1100 will (if available) provide priceable units to the possible solutions (i.e., level 4 nodes) that will then be used to build the next level (i.e., level 5 nodes) of the solution tree 1500. Subroutine 1100 is ~~repeated~~ repeatedly called and incrementally adds possible solutions that are below the current threshold cost. However, once subroutine 1100 has finished, it is then possible that a partial solution has been added that will provide a best fare, so routine 700 would then call the next subroutine 1200 for building level 5 of the solution tree 1500.

Amend the paragraph beginning on page 25, line 19, to read as follows:

A26

FIGURE 12 adds the final trip information to any eligible partial solutions of the solution tree 1500 by assigning fares to the partial solutions, thereby creating complete solutions (leaf a26 cont.

nodes in solution tree 1500). Subroutine 1200 begins at block 1201 and proceeds to block 1205, which tries to find a fare to be added to a partial solution with defined priceable unit(s) and that will form a complete solution that will not exceed a threshold cost (in one exemplary embodiment, the same threshold is used in the calling routine). Block 1205 uses both the MCM 465 and the FSR 470 to determine which fares are least costly and available for assignment. The partial solutions with priceable unit assignments are retrieved from the priority queue 475. More specifically, the cost of a new partial solution is calculated using all known trip information and then by using the MCM 465 and FSR 470 to complete any unknown trip information with feasible or actual fares for priceable units respectively. In block 1105 this means that the user query, breakpoint(s), carrier(s), flight(s) and priceable unit(s) are known trip information, so the FSR 470 would supply minimum actual fare information to a new unique possible solution, but if all the fares had yet to be assigned for a trip, then MCM 465 would supply feasible fares as well. As each new piece of trip information is added to the partial solutions, the cost of the partial solution will be refined, and will possibly increase, thereby removing it from consideration if the increased cost exceeds the threshold. Next, decision block 1210 determines if a fare was found. If no fare was found in decision block 1210, processing continues to block 1299, where any complete solutions, to which a fare has been added, are returned to the calling routine. It is at this point that the best fare routine 700 will have to determine if it has reached an ending condition (sufficient complete solutions, exhausted all possible solutions, or times out).

Amend the paragraph beginning on page 26, line 8, to read as follows:

a27

Otherwise, if a fare was found in decision block 1210, then a new partial solution is created by adding a fare (retrieved from the FRS 470) for which a fare has not been assigned in block 1215 and adding it to the priority queue 475. Next in block 1220 the fare rules, retrieved from the FSR 470, are applied to determine if any pricing changes need to be applied or if the fare is invalid. Processing continues to decision block 1225 where a determination is made whether the fare assigned in block 1215 to the new partial solution is an invalid fare. If in decision block 1225 it is determined that the new partial solution contains an invalid fare, then the new partial solution is discarded and removed from the priority queue 475. Processing then returns to block 1205 to search for any more partial solutions in which a fare may be added. If in decision block 1225 it is determined that it is not conclusively an invalid fare, then processing returns to block 1205 to search for any more partial solutions in which a fare may be added.

A27 Cont

Amend the paragraph beginning on page 26, line 21, to read as follows:

A28

Similarly to subroutines 800, 900, 1000 and 1100, subroutine 1200 is used to populate level 5 of the solution tree 1500. However, unlike subroutines 800, 900, 1000 and 1100, repeated calls to subroutine 1200 will (if available) provide the complete solutions that will be used to build the final level of the solution tree 1500. Subroutine 1200 is ~~repeated~~ repeatedly called and incrementally adds complete solutions that are below the current threshold cost. Once subroutine 1200 has finished, it is then possible to determine in routine 700 if an ending condition has been met. Ideally in routine 700 it will be determined that sufficient complete solutions have been found, in which case no more calls to subroutines 800, 900, 1000, 1100 or 1200 are needed and the best fares have been found (or in rare circumstances, no fares were found, or only a limited number of fares due to a time out condition).

Amend the paragraph beginning on page 26, line 32, to read as follows:

A29

As can be seen from the above discussion of the subroutines in FIGURE 7-12, each addition of trip information is handled in a similar manner. Each subroutine creates new partial solutions (complete solutions in the case of subroutine 1200), which are added to the solution

a29 Cont

tree 1500 in the priority queue 475 as they are created. By placing these solutions in the priority queue 475, ~~when~~ where they are later examined for consideration as a possible lowest cost solutions~~containing~~, only the most likely candidates for the best fares are examined first because the priority queue 475 is designed to provide the possible solutions in a lowest cost first manner as they are dequeued. Further, assuming that the best fare routine 700 was not interrupted, no partial solutions should ever be removed from the priority queue. Specifically, if all possible combinations of trip information were tried, then all nodes should be complete solutions. ~~One~~ On the other hand if a sufficient number of complete solutions were found in decision block 715, then by removing that number of complete solutions from the priority queue there should still be no ~~partials~~ partial solutions removed, as the complete solutions are necessarily lower cost (having beaten the threshold, while the partial solutions did not), and there are a sufficient number of complete solutions.

Amend the paragraph beginning on page 27, line 14, to read as follows:

a30

It is important to note that the best fare routine 700 is one possible best fare routine of ~~if~~ the present invention[[, in]]. In another exemplary embodiment of the present invention, possible solutions in a solution tree are examined both "forward" and "backward". FIGURE 13B illustrates a number of "breakpoints forward" 13A-E and "breakpoints backward" 13V-Z on either side of an imaginary "middle" 1355 of the trip. In this embodiment, the solution tree 1500 is built by examining flights both from the origin (forward), but also from the destination (backward) and once likely partial solutions are determined for both the forward portion and the backward portion, ~~then~~ they are combined as partial solutions 1360 and 1370. This forward and backward partitioning could be applied at each level of the best fare routine, such that FIGURES 7-12 would include examinations forward and examinations ~~back~~ backward when partitioning and adding partial solutions. One benefit of this "forward and backward" processing

A30 Cont'd

is a yet further decomposition of the problem of finding solutions ~~into~~ to smaller problems, such that it is better suited to recursive and/or parallel processing. In particular, in one actual embodiment of the present invention, this decomposition has allowed a best fare routine to be run on personal computers rather than on much more expensive mainframe computers.